AF/IFW

Attorney Docket No.: 944-1.70-2
Serial No.: 10/781,325

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

First named inventor: Jussi Piispanen

Serial No.: 10/781,325

Filed: Feb. 17, 2004

Title: METHOD AND APPARATUS FOR SYNCHRONIZING HOW DATA IS STORED IN DIFFERENT DATA STORES

Group Art Unit: 2152

Examiner: Kenny S. Lin

MAIL STOP APPEAL BRIEFS--PATENTS
COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450

FURTHER AMENDED BRIEF FOR APPELLANTS

IN RESPONSE TO NOTICE OF NON-COMPLIANT APPEAL BRIEF

Sir:

This is a further amended brief for an appeal from an Office Action mailed 28 August 2006, made final, to which applicant filed a request for reconsideration and in response received an Advisory Action, mailed 14 Nov. 2006, maintaining the rejections, and further in response to a Notice of Non-compliant appeal brief, mailed 28 March 2007, noting that the appeal brief as originally filed did not indicate in the statement of status of claims, which claims are being appealed, and further in response to a second notice of non-compliant appeal brief indicating that the "copy of the appealed claims should not include indicators." This amended brief is believed compliant.

This brief follows a timely filed Notice of Appeal, mailed to the Office on 28 November 2006.

For all of the reasons given below, it is the belief of the undersigned that the claims of the application do distinguish the

invention from the art relied on by the Examiner. Nevertheless, the undersigned is always willing to discuss possible amendments to any claims to clarify or resolve any issues related to claim interpretation that may remain after the Examiner has reviewed applicant's brief. The Examiner is strongly encouraged to call the undersigned to discuss making any such amendments.

Table of Contents

I. THE REAL PARTY IN INTEREST

The real party in interest is Nokia Corporation, having a principal place of business at Keilalahdentie 4, FIN-02150 Espoo, Finland.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF CLAIMS

Claims 1-6 and 8-30 are all the pending claims, and all stand rejected. Claim 7 is canceled.

Of the pending claims, only claims 1, 15 and 25 are independent.

The rejections of all the pending claims, namely 1-6 and 8-30, are appealed, although only the rejections of the independent claims 1, 15 and 25 are argued.

IV. STATUS OF AMENDMENTS

No amendments have been filed since the mailing of the final Office action.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Applicant provides herewith a concise explanation of the subject matter defined in each of the independent claims involved in the appeal, referring to the specification by page and line number, and to the drawings, by reference characters. No dependent claims are involved in the appeal.

The invention has to do with synchronizing the contents of data stores, typically data stores hosted by different devices. In the terminology used in the application, a data store contains data units contained in folders (which may be nested) according

to some arrangement, so that the data units have what is called a data structure or a directory structure. See page 15, lines 1-13.

Synchronizing two data stores requires, in general, synchronizing both the data units (e.g. files in a directory) and the data structures (e.g. the arrangement of folders in a directory structures), i.e. how the folders containing the data units are arranged with respect to each other, so that both data stores have all the same folders organized in the same way. A typical synchronizing would allow for the use of data stores in two different locations. The synchronization then might arrange both data stores to have all the most recent changes made to both data stores in respect to both the folder organization and the data units.

As explained at page 2, beginning line 10, SyncML (synchronization markup language) is an open industry standard for a common language for universal synchronization of remote data (i.e. data items stored in different equipment and so in different data stores) and personal information across multiple networks, platforms and devices. SyncML is being developed under the so-called SyncML Initiative. With SyncML, data items/ data units, but not yet data structure/ directory structure, can be synchronized on different devices connected via one or more interconnecting networks. At page 6, lines 8-11, the application explains that according to the prior art for SyncML, if a user changes folders on a device, SyncML cannot be used to synchronize the data store on the device with a corresponding data store being maintained on another device; the prior art for SyncML allows only synchronizing with respect to changes in data units.

As explained at page 6, line 12, a SyncML message is a nested structure, and one or more SyncML messages can be associated with what is called a SyncML package. The SyncML Message is an individual XML document consisting of one or more

elements each of one or more element types. The document consists of a header, specified by the **SyncHdr** element type, and a body, specified by the **SyncBody** element type. The SyncML header specifies routing and versioning information about the SyncML Message. The SyncML body is a container for one or more SyncML Commands. The SyncML Commands are specified by individual element types. The SyncML Commands act as containers for other element types that describe the specifics of the SyncML command, including any data or meta-information. SyncML defines *request commands* and *response commands*. Request commands include, for example: add (a command that allows the originator to ask that one or more data units be added to data accessible to the recipient); and alert (allowing the originator to notify the recipient of a condition). According to SyncML, these commands are included in a data element of a protocol command element. (Fig. 2 shows a protocol command element of a SyncML container, where the protocol command element includes both data elements and non-data elements.)

Claim 1 is to a method in which one of two devices sends a message to the other for synchronizing a data store on one and a data store on the other. The two devices are shown e.g. in Fig. 1, as client device 11 and server device 12. The device recited in claim 1 as sending the message can be either the client device or the server device. Depending on which it is, the message is shown in Fig. 1 as the client SyncML message (see the data flow emanating from the client sync. agent 11b) or the server SyncML message (see the data flow emanating from the server sync, agent 12b).

Claim 1 recites a first device (client 11 or server 12 or Fig. 1) preparing a message (the client or server SyncML message of Fig. 1, and see also Fig. 1, showing the structure of the message) including information indicating a folder (an exemplary folder being shown in Fig. 4 as folder F1 or folder F1-1 or

folder F1-2, and so on) useable for storing data in a data store (client data store 11c or server data store 12c) of the first device, with the message including a header and a body (see header 22 and body 23 of the SyncML message/ container 21 of Fig. 2), each in turn comprising one or more elements (which can be nested, as explained above re: page 6, line 12, i.e. the header is itself an element, and the body includes elements that includes elements, such as protocol command element 24 including non data element 25, and so on), with the body elements useable for providing commands in connection with synchronizing the first data store with respect to a data store in another device (the protocol command element 24 and its nested elements 25, 26 and 26a, and so on) and also useable for conveying data from the data store (via data elements 28 and 29, and so on), and the information indicating the folder of the data store uniquely identifies the folder and is placed in the message in an element different from where data of the data store is placed or would be placed if included in the message (i.e. in the non data element container 25).

As explained at page 17, beginning line 7, the invention uses the same commands for changing folders as are used to change data units; thus, changes to folders are made by issuing messages conveying operational elements (Protocol Command Elements), such as Sync, Add, Replace and Delete, in which the affected folders are referenced within what are here called Data Identification Elements, meaning either TARGET or SOURCE elements, external to DATA elements. To make a change to a data unit, SyncML according to the prior art calls for a message referencing the data unit within a DATA element nested in an operational element. The application then explains beginning page 17, line 29, in reference to Fig. 2, that according to the invention folders (and thus the data organization in terms of directories) affected in the course of synchronizing the data stores 11c 12c are identified or indicated in the LocURI elements 26a 27a of the

Source element 26 and the Target element 27 of a protocol command element 25 (such as replace) external to (i.e. not contained in) a data element 28 29 (i.e. a particular type of data description element). Both the Target and Source elements are here called *Data Identification Elements*, to stand in contradistinction to the *data elements* 28 29 that contain the data units.

Claim 15 is to a device (client 11 or server 12 of Fig. 1), comprising: a data store (11c or 12c of Fig. 1), for storing folders (F1, F1-1, … of Fig. 4) useable for storing data; and means (client sync. agent 11b or server sync. agent 12b) for preparing a message (client or server SyncML message of Fig. 1) including information (non-data element 25 of Fig. 2) indicating a folder in the data store, wherein the message includes a header (22 of Fig. 2) and a body (23 of Fig. 2), each in turn comprising one or more elements (header 22 is an element, and body 23 is shown as nestably including elements 24, 25, and so on), with the body elements useable for providing commands (Protocol Command Element 24 of Fig. 2) in connection with synchronizing the data store with respect to another data store in a second device and also useable for conveying data (data element 28 and 29 and so on) from the data store; wherein said information indicating the folder of the data store uniquely identifies the folder, and the device is configured to place the information in the message in an element (non-data element 25 of Fig. 2) different from where data of the data store is placed (data element 28 or 29 and so on of Fig. 2) or would be placed if included in the message.

Claim 25 is also to a device (client 11 or server 12 of Fig. 1), comprising: a data store (11c or 12c of Fig. 1), for storing folders (F1, F1-1, … of Fig. 4) useable for storing data; and a synchronization agent (client or server synchronization agent 11b or 12b), for preparing a message including information indicating a folder in the data store, wherein the message includes a header and a body, each in turn comprising one or more elements, with

the body elements useable for providing commands (Protocol Command Element 24 of Fig. 2) in connection with synchronizing the data store with respect to another data store in a second device and also useable for conveying data from the data store; wherein said information indicating the folder of the data store uniquely identifies the folder and the device is configured to place the information in the message in an element (non-data element 25 of Fig. 2) different from where data of the data store is placed (data element 28 or 29 and so on of Fig. 2) or would be placed if included in the message.

Thus claims 15 and 25 recite limitations corresponding to those of claim 1.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The following grounds of rejection are to be reviewed:

the rejections of claims 1, 15 and 25 under 35 USC §103 as being unpatentable over Applicant Admitted Prior Art (AAPA) in reference to SyncML Initiative (including standards and specifications for SyncML, SyncML Representation Protocol, and SyncML Sync Protocol, SyncML Device Management Protocol).

VII. ARGUMENT

To reject the claims, the Examiner asserts that:

> ... session 4, Sync Initialization of SyncML Sync Protocol of SyncML Initiative shows the use of Alert command (e.g. element) for authentication and indication of which database to be synchronized and Put and Get commands (e.g. different element from Alert command) for conveying the data. This clearly shows the claimed limitation of "information indicating the folder of the data store uniquely identifies the folder and is placed in the message in an element different from where data of the data store is placed or would be placed if included in the message."

Applicant respectfully submits that *first*, a "database" cannot be understood as a "folder" as that term is used in the claims, and *second*, irrespective of whether a folder as used in the claims encompasses a database, SyncML teaches using the Alert command (and the Put and Get commands) in a data element portion of a SyncML message, not in a non-data element as required by the claims (on account of the recitation, "wherein said information indicating the folder of the data store uniquely identifies the folder and is placed in the message in an element different from where data of the data store is placed or would be placed if included in the message").

As to the *first* point, the rejection by the Examiner requires construing the claims so that a "database," as that term is used in SyncML Sync Protocol of SyncML Initiative (hereinafter SyncML), is the equivalent of a "folder" as the latter is used in the claims. Applicant respectfully submits that "database" as used in SyncML clearly refers to an entire collection of folders (all files in all folders in case of a database organized as folders of files) or an entire set of data records, etc., or files of such. In contrast, the term "folder" as used in the claims must be construed as one element of such an entire collection because that is how the term "folder" is used in the application. The Federal Circuit has explained that terms in the claims are to be interpreted based on how the terms are used in the specification. In *Phillips vs. AWH Corp.*, 415 F.3d 1303, 75 USPQ.2d 1321 (Fed. Cir. 2005), an *en banc* decision, the Court explained that:

> [T]he specification is always highly relevant to the claim construction analysis. Usually, it is dispositive; it is the single best guide to the meaning of a disputed term.

The Court further explained:

> That starting point [for understanding a claim term] is based on the well-settled understanding that inventors are typically persons skilled in the field of the invention and that patents are

> addressed to and intended to be read by others of skill in the pertinent art. … Importantly, the person of ordinary skill in the art is deemed to read the claim term not only in the context of the particular claim in which the disputed term appears, but in the context of the entire patent, including the specification.

Applicant respectfully points out that the application explains at page 4, ll. 18-25:

> ... what is called a folder in various operating systems (such as Windows, available from Microsoft Corporation) is to be understood by the term folder as used here, but so is a record in a table of a relational database, since a record in such a table includes fields, which are data units. Even such a field can be considered a folder, since a field contains characters, numbers, or other elements that can be considered data units, and thus each field is a folder.

See also page 5, beginning line 22, where the application explains:

> Also, since as has already been explained, a folder can indicate for example a record in a table of a database or even a field within such a record, the terminology directory structure or data structure should also be understood to encompass even a database structure (when a folder is a record) or a record structure (when a folder is a field).

Thus, a folder is clearly used in the application to indicate only one folder in a set of folders (the set of folders then being a directory structure), or one record in a set of records (the set of records then being e.g. a database), or one field in a set of fields (i.e. one field in a record, the record then being, in that instance, a data structure as that term is used in the application). Although what is a folder/ container of data units, and what is a data unit can vary, depending on the context, what is clear is that a folder is never, in any context, to be understood as all data units in a data store, but is instead one element of the data structure of the data store of data units. As explained, the invention is aimed at enabling synchronizing the data structure, and so it must be possible to

indicate not the entire data structure, but a particular element of it. Construing a folder to be an entire database would make the claimed invention of no help in achieving the object of synchronizing the data structure.

Per the Court in *Phillips vs. AWH Corp.* then, applicant respectfully submits that the Office cannot interpret a "database" to be a "folder."

As to the *second* point in this regard, as pointed out in the application at page 17, ll. 18-21, to make a change to a data unit, SyncML "calls for a message referencing a data unit within a DATA element nested in an operational element." So with SyncML, to reference a data unit the DATA element must indicate the folder where the data is (or is to be placed, in case the data/folder is new), since SyncML provides no other basis for indicating the folder. (So with SyncML, it is data units that are synchronized, not the data structure, i.e. e.g. one cannot use the prior art SyncML to create an empty folder, and one cannot refer to a particular folder except in respect to a particular data unit, as in a path statement.) Claims 1, 15, and 25 on the other hand require that the folder for such data (or a new folder) be indicated in "an element different from where data of the data store is placed or would be placed if included in the message." The advantage provided by the invention is explained at page 17, line 21:

> It is the use of a reference to affected folders external to the DATA element (i.e. a data description element) that gives the invention advantages over other possible arrangements for referring to affected folders (namely, not having to duplicate code enabling SyncML parsing by each application, and not having to include in each sync agent code for interpreting the data units of each different application).

Applicant respectfully submits that the invention is thus clearly distinguished from AAPA in that the invention indicates a folder in a non-data element where a folder is one element of a

data structure, and that the difference makes a difference, i.e. not having to include in each sync agent, code for interpreting the data units of each different application.

For the reasons given, then, applicant respectfully submits that the rejections of claims 1, 15 and 25 under 35 USC §103 are error, and so therefore are the rejections of all the other pending claims, namely 2-6 and 8-14 and 16-24 and 26-30

Respectfully submitted,

19 July 2007
Date

James A. Retter
Registration No. 41,266

WARE, FRESSOLA, VAN DER SLUYS
& ADOLPHSON LLP
755 Main Street, P.O. Box 224
Monroe, CT 06468-0224

tel: (203) 261-1234
Cust. No.: 004955

VIII. CLAIMS APPENDIX

The following are the claims involved in the appeal.

1. A method, comprising:

a first device preparing a message including information indicating a folder useable for storing data in a data store of the first device, wherein the message includes a header and a body, each in turn comprising one or more elements, with the body elements useable for providing commands in connection with synchronizing the first data store with respect to a data store in another device and also useable for conveying data from the data store; and

the first device sending the message to the other device;

wherein said information indicating the folder of the data store uniquely identifies the folder and is placed in the message in an element different from where data of the data store is placed or would be placed if included in the message.

2. A method as in claim 1, wherein the element where the information indicating the folder is placed in a field of the message.

3. A method as in claim 1, wherein data of the data store is placed or would be placed in a data element of the message.

4. A method as in claim 3, wherein the data element is a data element of a protocol command element.

5. A method as in claim 1, wherein the information indicating the folder is included in a non-data element of the message.

6. A method as in claim 5, wherein the non-data element is a non-data element of a protocol command element.

7. Canceled.

8. The method of claim 1, wherein a data identification element is contained in a protocol command element in the message, and the protocol command element in combination with the data identification element indicates the folder of the data store of the first device.

9. The method of claim 1, wherein a data identification element is included in the message and the information indicating the folder of the data store of the first device is provided in the data identification element.

10. The method of claim 1, wherein the first device functions as a client in a client-server protocol and the second device as a server in the client-server protocol.

11. The method of claim 1, wherein the first device functions as a server in a client-server protocol and the second device as a client in the client-server protocol, and in preparing the message the first device is responsive to a client message from the second device and includes resolving any conflicts posed by the client message in respect to the data store of the first device.

12. The method of claim 1, wherein the data in the data stores are used for device management by applications hosted on the devices.

13. The method of claim 1, wherein the data in the data stores are used as user data by applications hosted on the devices.

14. A computer program product comprising: a computer readable storage structure embodying computer program code thereon for execution by a computer processor, with said computer program

code characterized in that it includes instructions for performing the steps of the method of claim 1.

15. A device, comprising:

a data store, for storing folders useable for storing data; and means for preparing a message including information indicating a folder in the data store, wherein the message includes a header and a body, each in turn comprising one or more elements, with the body elements useable for providing commands in connection with synchronizing the data store with respect to another data store in a second device and also useable for conveying data from the data store;

wherein said information indicating the folder of the data store uniquely identifies the folder, and the device is configured to place the information in the message in an element different from where data of the data store is placed or would be placed if included in the message.

16. A device as in claim 15, wherein the device is either a wireless communication terminal or a wireline communication terminal.

17. A device as in claim 15, wherein the device is configured to function as a client in a client-server model.

18. A device as in claim 15, wherein the device is configured to function as a server in a client-server model, and further comprises means for receiving a request to synchronize from the second device, and for then sending the message in response to the request to synchronize.

19. A device as in claim 15, further comprising means for receiving from the second device a message including information indicating a folder in the other data store, wherein the message

includes a header and a body, each in turn comprising one or more elements, with the body elements useable for providing commands in connection with synchronizing the other data store with respect to the data store in the device and also useable for conveying data from the other data store, and wherein the device is configured to function as a server in a client-server model and includes means for resolving conflicts posed by the message.

20. A device as in claim 15, wherein the data in the data store is used for device management by applications hosted on the device.

21. A device as in claim 15, wherein the data in the data store is used as user data by applications hosted by the device.

22. A system, comprising a device according to claim 15, and also comprising the second device hosting the other data store.

23. A system as in claim 22, wherein the device is configured to function as a server in a client-server model and the second device functions as a client in the client-server model.

24. A system as in claim 23, wherein the device is configured to send the message to the second device in response to a request sent by the second device to synchronize to the second device.

25. A device, comprising:

a data store, for storing folders useable for storing data; and

a synchronization agent, for preparing a message including information indicating a folder in the data store, wherein the message includes a header and a body, each in turn comprising one or more elements, with the body elements useable for providing commands in connection with synchronizing the data store with

respect to another data store in a second device and also useable for conveying data from the data store; and

wherein said information indicating the folder of the data store uniquely identifies the folder and the device is configured to place the information in the message in an element different from where data of the data store is placed or would be placed if included in the message.

26. A device as in claim 25, wherein the device is either a wireless communication terminal or a wireline communication terminal.

27. A device as in claim 25, wherein the device is configured to function as a client in a client-server model.

28. A device as in claim 25, wherein the device is configured to function as a server in a client-server model, and further comprises a synchronization adapter for receiving a request to synchronize from the second device, and for then sending the message in response to the request to synchronize.

29. A device as in claim 25, wherein the synchronization adapter is configured to receive from the second device a message including information indicating a folder in the other data store, wherein the message also includes a header and a body, each in turn comprising one or more elements, with the body elements useable for providing commands in connection with synchronizing the other data store with respect to the data store in the device and also useable for conveying data from the other data store, and wherein the device functions as a server in a client-server model and the synchronization agent is configured to resolve conflicts posed by the message.

30. A device as in claim 25, wherein the data in the data store

is used for device management by applications hosted on the device.

IX. EVIDENCE APPENDIX

No evidence has been submitted under Rules 1.130, 1.131, or 1.132 and relied on by appellant in the appeal, nor is there any other evidence entered by the examiner and relied up on by appellant in the appeal.

X. RELATED PROCEDINGS APPENDIX

There are no and have been no related proceedings, and so there are no corresponding decisions rendered by a court or the Board in any related proceeding.